Chapter 47
The Impact of Knowledge Sharing Platforms in Distributed Requirements Engineering Scenarios: A Systematic Review

Christian Sillaber and Ruth Breu

Abstract Knowledge sharing platforms promised to solve a number of challenges that complicate requirements engineering activities in globally distributed software engineering efforts. A systematic review of empirical studies of the impact of such knowledge sharing platforms up to and including 2011 was conducted. The selected search strategy identified 511 studies, of which 17 were identified as suitable empirical studies. The studies were evaluated according to the presented knowledge sharing platform, the requirements engineering setting as well as to the reported impact on the requirements engineering processes. This systematic review investigates what is currently known about the evidence for positive and negative impacts of knowledge sharing platforms in requirements engineering processes. The main finding of this systematic review is that currently little empirical data on knowledge sharing platforms and their use in requirements engineering processes is available and further empirical research is necessary.

47.1 Introduction

During the last two decades, the management, elicitation and analysis of software requirements have evolved from being concentrated at a single development team location to being geographically distributed across multiple countries and timezones. A variety of business reasons including cost savings [1], the increased availability of skilled labour [2] as well as the possibility to develop software

C. Sillaber (✉) · R. Breu
Institute of Computer Science, University of Innsbruck,
Technikerstrasse 21a 6020 Innsbruck, Austria
e-mail: christian.sillaber@uibk.ac.at

R. Breu
e-mail: ruth.breu@uibk.ac.at

L. Uden et al. (eds.), *The 8th International Conference on Knowledge Management in Organizations*, Springer Proceedings in Complexity,
DOI: 10.1007/978-94-007-7287-8_47,

products 24/7 [3] has led to an increased difficulty of quickly sharing knowledge on software requirements.

Many remedies for improving the knowledge exchange between stakeholders have been suggested, from the re-use of established mediums as telephone and video conferencing to specifically developed collaborative knowledge sharing (KS) platforms. However, though there are many tools available for requirements engineers, little is known about their use in practice and what their effects are.

This systematic review seeks to evaluate, distill and present the empirical findings on the impact of KS platforms in requirements engineering (RE) to date, and provide an overview on the different research topics and their findings. We believe that this review will help the scientific community to identify areas and opportunities where research is lacking.

The article is organized as follows: In Sect. 47.2, we give an overview of RE with a focus on KS challenges specific to distributed environments. The methods used for this review are described in Sect. 47.3. Section 47.4 reports the findings of this review and presents an overview of the studies and the reported results. The benefits and limitations, strengths and weaknesses and implications for research are presented in Sect. 47.5. Section 47.6 concludes and provides recommendations for further research on the use of KS platforms for RE.

47.2 Background-Requirements Engineering and Knowledge Sharing

As part of the globalization efforts prevalent in the last decade(s), software development teams have increasingly become geographically distributed [3–5]. This trend of adding more (Global Software Development, GSD [6]) or less (Distributed Software Development, DSD [7]) distance between team members has created additional challenges for requirements engineers to share knowledge across team members [5, 8–10].

We first describe the field of RE and recent challenges that make the development and use of KS platforms inevitable. We then summarize previous works on the challenges of KS in RE processes and justify the need for this review. Lastly, we state the research questions that motivated the review presented in this paper.

Requirements engineering: Requirements engineering is a discipline strongly linked to the success and failure of software development processes regardless their size, available resources or the structure of the development team [11–14]. Beneath technical activities, RE involves a plethora of challenging non-technical activities including negotiation, analysis and management tasks [15]. These activities, requiring active communication and the exchange of knowledge can be found in all phases of RE processes and their importance has been shown in numerous works, e.g. [5, 16–21]. Team members need communication to coordinate their activities and share knowledge as documentation often becomes obsolete and relevant knowledge often only resides with people [22]. The importance of KS in large, often geographically distributed, multicultural development teams comprised of diverse software and RE roles, that are prevalent today [10, 23], is well acknowledged in literature, e.g. [3, 5, 9, 24, 25].

Knowledge sharing challenges in requirement engineering: Distributed software development, in particular the associated RE processes are fraught with challenges [3]. In literature the five main challenges are *inadequate communication*, *cultural and language differences*, *process differences*, as well as *technical differences* [3, 5, 23]. The reasons for inadequate communication are manifold ranging from time-zone difficulties to inadequate means of expression [3, 9]. In the case of multinational requirement engineering teams, cultural and language difficulties arise from different stakeholder backgrounds [10, 26]. Besides differences in human culture, different corporate cultures pose additional challenges, as they may vary from one site to another, as language might [3]. Inadequate or incompatible processes often lead to challenging documentation and traceability problems [23]. While technical difficulties such as differences in bandwidth or available hardware resources might be solved in the foreseeable future, different platforms and document formats pose additional challenges. Often, these problems are interwoven and therefore challenging on the technical layers as well as the process and knowledge layer. For instance [3] reports that *"… remote practitioners are unable to hold effective discussions on requirements. Since existing requirements management tools do not provide rich support for collaboration, teams typically use these tools only as a shared requirements repository and hold all discussions outside of the tool in email, chats or phone-calls…"*. The variety of channels requires expensive cognitive switches and are also hard to track centrally leading to decreased trace-ability. Besides trace-ability issues, knowledge of stakeholders as well as their understanding of the requirements is hard to quantify and even harder to share in global scenarios as the individual interpretation of requirements might change according to the cultural or organizational background [3, 10, 25].

Objectives of this review: Knowledge sharing, and KS platforms have attracted huge interest from both software and RE research and practice. However, to the best of our knowledge, no systematic review of the impact of KS platforms in requirements engineering has previously been published. The existing works presented in the previous section only partially cover the evaluation of real-world solutions to address KS issues and challenges in RE practice. Furthermore, they do not include a systematic assessment of the actual impact of KS platforms in real world RE environments. The objective of this systematic review is to answer the following research questions: **(1)** What is currently known about the positive and negative impact of KS platforms in requirements engineering? **(2)** What is the strength of the evidence in support of these findings? **(3)** What are the implications of these studies for the requirements engineering industry and the research community?

47.3 Review Method

Following the methodology for integrating diverse software engineering study types, as presented in [27], as well as the established method of systematic review, we conducted the review in six successive stages: development of review protocol, the identification of inclusion and exclusion criteria, the identification of relevant studies, critical appraisal, data extraction and synthesis (cf. [27]).

(1) Protocol development: Following the protocol presented in [27], an Org-Mode protocol [28] for the systematic review was developed and aligned with [29]. In this protocol, we specified the research agenda and questions, search strategy, exclusion, inclusion criteria as well as data extraction and synthesis methods.

(2) Inclusion and exclusion criteria: We deemed articles presenting empirical data on the usage of KS platforms and their impact on requirements engineering processes in both academic and professional environments eligible for inclusion. Qualitative and quantitative research studies published in the English language until 2012 were included (not restricted to any specific model, methodology, or technology or the reported outcome). Furthermore, studies reporting only on the impact of KS platforms of specific (sub-) aspects were deemed eligible (e.g. only requirements analysis or only negotiation phase)-independently from the RE methodology. We excluded studies if their focus was not on the use of KS platforms in (software) requirements engineering. Also, studies that did not present empirical data or were mere "lessons-learned" papers were excluded. Papers describing the impact of KS platforms in non-software development processes (e.g. new product development) were excluded. As this review is concerned with the impact of KS platforms in RE as a whole, studies that focused on single techniques or aspects, such as requirements recommender systems in KS platforms were excluded.

(3) Data sources and search strategy: The search strategy included automated and manual searches in electronic databases. The following electronic databases were queried: ACM Digital Library, IEEE Xplore, ISI Web of Science, ScienceDirect—Elsevier, SpringerLink, Wiley Inter Science Journal Finder, Jstom, and EBSco Host.

Figure 47.1 shows the structure plan for the systematic review process and the number of papers identified at each stage. In the first stage, the titles, abstracts, general meta-data and keywords of the articles identified were extracted from the corresponding electronic database. In the electronic database, we searched using the following query: knowledge NEAR sharing AND (software OR platform) AND requirement* NEAR engineering.

In case an electronic database did not support the NEAR syntax, an AND query was used instead and the results were manually inspected for word proximity. From the retrieved results, we automatically excluded editorials, article interviews, reviews, discussions, tutorials, and poster sessions. This search strategy resulted in a total of 511 hits from all electronic databases that included a set of automatically identified 475 unduplicated citations.

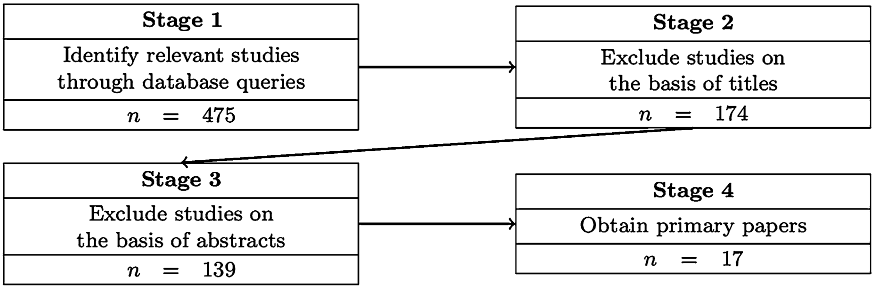
Stage 1
Identify relevant studies through database queries
$n = 475$
Stage 2
Exclude studies on the basis of titles
$n = 174$
Stage 3
Exclude studies on the basis of abstracts
$n = 139$
Stage 4
Obtain primary papers
$n = 17$

Fig. 47.1 Stages of the study selection process

(4) Citation management, study retrieval, and inclusion decisions: Relevant citations from stage 1 (n = 475) were automatically retrieved and stored with the aid of JabRef, a citation management software. There we recorded the retrieval source, decision, status and eligibility decision. For each subsequent stage, separate BiBTex databases were established.

In stage 2, we manually went through the titles of all database entries that resulted from the previous stage, to determine their relevance to the systematic review. At this stage, we excluded studies that were clearly not about the impact of KS platforms in requirements engineering, independently of their nature. As an example, because our search strategy was rather broad, including the terms "requirements" and "knowledge sharing", we got several matches about requirements for KS platforms as well as requirements for ontologies covering KS. Also studies covering requirements engineering not related to the development of software were dismissed. Articles with titles that indicated clearly that the articles were outside the scope of this systematic review were excluded. However, titles are not always clear indicators of what an article is about (similarly, e.g. [27, 30]). Also the use of the term "requirements engineering" was sometimes rather ambiguous. In these cases, the articles were included for review at the next stage. At this stage, 301 articles were excluded (n = 174). In the third stage, the abstracts of the remaining papers were evaluated according to the previously described scheme and 35 studies were excluded (n = 139).

Each of the 139 studies that remained after the previous stage was assessed, according to criteria presented in [27]. The therein developed criteria cover three main issues pertaining to quality that the authors believe to require consideration when appraising the studies identified in the review: *Rigor*: Has a thorough and appropriate approach been applied to key research methods in the study? *Credibility*: Are the findings well-presented and meaningful? *Relevance*: How useful are the findings to the requirements engineering community?

Based on these criteria and the suggested evaluation questions of [27], we developed the following assessment criteria as to whether: (1) The study reported empirical research on implementing KS platforms in RE projects in academia or industry, (2) the aims and objectives were clearly reported and whether the focus was on knowledge aspects or requirements engineering aspects, (3) there was an adequate description of the context in which the research was carried out (e.g. industry setting, RE methodology), and (4) the reported findings were sound and clearly presented with a justified conclusion. The 139 studies from the previous stage went through the final evaluation according to the above scheme and excluded according to the answer to question (1). Out of the 139 studies, 122 were excluded and 17 remained for the detailed data extraction and synthesis.

(5) Data extraction and synthesis of findings: After going through stages 1 to 4, the data from the 17 studies selected for this systematic review was extracted according to a predefined extraction form. During the data extraction, annotations in the papers were used to highlight relevant passages and research results. These passages were imported into the Org-Mode document for future reference. To synthesize the findings, the terminology from the studies was at first taken verbatim from them and then "translated" according to a common glossary. Also, a rough categorization of the author's terminology was applied to ensure that similar or the same KS platforms hidden behind "different" names or commercial vendors' names were identified correctly.

47.4 Results

We identified 17 studies that report on the impact of KS platforms in RE. We categorized the studies into five main groups according to the main activities of RE: analysis, specification, modeling, validation, and management.

We found that out of the 17 studies, 9 (52 %) focused on a globally distributed software development scenario. Offshore software development scenarios were studied in 2 (11 %) articles. Studies describing general software development scenarios without any specific mention of the team members' location come next with 3 (17 %) of the studies. The results are shown in Table 47.1.

As shown in Table 47.2, most studies (82 %) dealt with professional software developers. The remaining (17 %) were conducted in an academic setting (e.g. Software Development course). Observed team sizes varied largely between studies. No particular focus on countries regarding the location of the team members could be observed. Regarding the year of publication, we found no suitable studies on the impact of KS platforms in RE prior to 2006 (Publications

Table 47.1 Studies after type of software development scenario

Software development scenario	Number	Percent (%)
Global Software Development (GSD)	9	52
Offshore Software Development (OSD)	2	11
General software development	3	17
Other	3	17
Total	17	100

Table 47.2 Overview on primary studies and their setting

ID	Reference	Research method	SD method	Professional/Student
S1	[38]	Multicase	OSD (RUP)	Students
S2	[10]	Survey	GSD	Professionals
S3	[9]	Survey	–	Professionals
S4	[39]	Survey	GSD (Scrum)	Professionals
S5	[26]	Singlecase	GSD	Students
S6	[40]	Singlecase	SD	Professionals
S7	[3]	Singlecase	GSD	Professionals
S8	[23]	–	GSD	Professionals
S9	[41]	Singlecase	GSD	Professionals
S10	[42]	Singlecase	GSD	Professionals
S11	[6]	Singlecase	GSD	Professionals
S12	[43]	Singlecase	–	Professionals
S13	[44]	Singlecase	–	Professionals
S14	[1]	–	OSD	Professionals
S15	[22]	Singlecase	GSD	Professionals
S16	[32]	Survey	SD	Professionals
S17	[45]	Survey	SD	Students

per year: 2006: 2 (11 %), 2007: 5 (29 %), 2008: 2 (11 %), 2009: 2 (11 %), 2010: 4 (23 %), and 2011: 4 (23 %)). In the following, citations starting with an uppercase "S" denote surveyed papers.

47.4.1 Challenges and Obstacles Addressed by Knowledge Sharing Platforms According to the Surveyed Literature

With the increasing number of integration projects and changing team setups the transfer of development knowledge becomes increasingly important [S4]. Studies described several obstacles that we tried to group according to the following scheme:

Inadequate communication: Several studies, e.g. [S4, S5, S14, S15, S16] report that the lack of face-to-face communication (e.g. "Communication gap" [S14]) is still a major inhibitor in RE. The inadequate measures available to early address communication problems often lead to costly problems and difficulties that spread over the different software development phase [3, 31, 32]. Inadequate communication is often due to a lack of common understanding of concepts. E.g. [S5] reports on the importance of commonly shared ontologies. Inadequate communication also often entails a lack of global project awareness as well as difficulties to transfer results between different business domains [S4, S14] (cf. "Interpretation" in [S8]). Trace-ability and the exchange of prioritization knowledge still remains as a huge challenge in knowledge sharing practice [S8]. Few reported cases have addressed the challenge of discussing and tracking non-functional requirements [S5, S8].

Cultural diversity [S1, S2, S5, S7, S8, S11, S14, S15, S17] mention that, particularly due to recent outsourcing trends to Non-Western countries, cultural obstacles arise that must be addressed. According to [S7], a distinction can be made between differences in the individuals' culture as well as the corporate culture, the individual is part of. A problem highly related to the cultural diversity of stakeholder is that explicit knowledge on participating stakeholders is hard to share [S8]. In [S8] it is also reported that knowledge on different markets and customers varies largely between stakeholders.

Time difference: A rather frequently discussed problem, e.g. [S1, S2, S5, S7, S8] that is due to the fact that geographic separation of team members also often means that the time-zones of the team are also separated, and often even do not overlap. Different time zones also negatively influence the stakeholders' ability to quickly share information thus further worsening the communication problem. [S14] reports that adequate mechanisms for document maintenance and synchronization must be used.

47.4.2 Impact on Requirements Engineering

The overall goal of using KS platform to support KS activities in RE has been described by researchers: *"The aim of these knowledge intensive interactions, often embedded in requirements elicitation, analysis and verification activities, is to collaboratively transform the initial uncertain and ambiguous understanding of the domain problem into an application concept, consistent system requirements, and ultimately a software application that can be used by the target organization."* [S3]

[S4] observed that the majority of the participants view knowledge documentation as being important or very important. Regarding the scope of KS platforms used, [S4] observed that 47 % of surveyed agile developers use KS software supporting online discussions. Evidence is presented in [S5] that the selection of KS platforms, in that case groupware tools, which are closer to the stakeholders cognitive style measurably improved the stakeholders' perception of communication during the requirements elicitation phase. Stakeholder using a tool that did not match the cognitive characteristics led to a worsened perception of communication [S5].

From a technological standpoint, [S6] reported that a bias towards wikis existed for two reasons: (1) not many people knew about them and (2) personal bias existed toward new technologies. (1) can be easily understated by the fact that [S6] is among the oldest studies (2007) and awareness of Wikis and their use has increased in the meantime [33–37]. Several studies address how KS platforms are introduced and adopted in companies to improve the requirements elicitation phase. The goal of [S5] was to develop and evaluate a framework to improve and understand communication during the requirements elicitation process. In [S14] a need for better trace-ability of requirements was established. In a similar vein, [S2] showed that it is important to identify stakeholder centric social networks in addition to purely managing communication responsibilities.

47.4.3 Reported Knowledge Sharing Platforms

The studies can be categorized according to whether they describe the results of (1) applying existing KS platforms in RE, (2) applying a KS platform developed by the authors in RE (3) analyzing KS platforms already used in industry settings. The established KS platforms described in type 1 studies are: e-mails, newsgroups, mailing lists, forums, electronic notice boards, shared whiteboards, document sharing, chat, instant messaging, wikis, and videoconferencing [S5, S7, S16]. The terminology for these technologies is used consistently across almost all studies. However no standardized definition of these KS platforms, in particular of "wikis", "document sharing" and "notice boards" could be found.

47.5 Discussion

This review identified 17 studies on the impact of KS platforms on RE. To the best of our knowledge, no systematic review exists in this domain. In this section, we address our research questions, starting by discussing the impact of KS platforms in requirements engineering, the strength of the evidence in support these findings as well as limitations and future implications for industry and research.

Impact of knowledge sharing platforms in requirements engineering: Studies addressing the introduction and adoption of KS platforms in RE processes fail to establish a unified picture of current practice. They provide sparse views into the experience of RE teams with KS platforms. For all the identified challenges (c.f. 4.1), KS platforms were reported to have a positive impact. However, little to none details on the actual impact and its quantification could be identified in the studies. Common agreement exists on the fact that KS platforms are required in any distributed RE setting. While KS platforms are often not explicitly named as such, there seems to be a common agreement on what KS platforms are. The reported impact was positive in all studies irregardless of the technical specifics of the KS platform, no negative reports could be identified. Apart from multitasking problems when using multiple KS platforms, no negative impacts were reported.

Strength of evidence and implications for research and practice: The findings of this review have a number of implications for research and practice. For research, the review shows a clear need for more empirical studies of KS platforms and their impact on RE. KS platforms seem to have become so ubiquitous in recent years that hardly anybody seems to question their use-fullness or wants to evaluate their impact empirically. In our opinion, much more structured research is necessary to understand how KS platforms affect RE and how to properly respond to the broad adoption of such platforms. We have also seen during the review that Wikis seem to be, with rare exceptions, the only general purpose answer. Therefore, research on (1) other KS platforms and (2) the various types of Wikis is highly warranted, as there does not seem to be a common agreement on what functionalities such a Wiki must provide to successfully support RE. Another challenge is to increase the quality of the reported research results. In order to increase the usefulness of research on KS platforms and their impact on RE, we believe that researchers in the fields of KS and RE should collaborate to determine a common research agenda. The most relevant evidence for practice from the systematic review is that there exists a general agreement that KS platforms in general have a positive impact on all stages of the RE process. However, it is almost impossible to offer advice besides "use a KS platform to improve your RE processes", as there is often no evidence reported in the studies. Also it is almost impossible to compare an arbitrary industry setting against the studies as most of them lack the necessary details to make a well informed comparison. Therefore, this review only provides an initial overview to practitioners that need to carefully compare existing KS approaches to their own needs. This review clearly shows the need for knowledge and requirements engineering researchers to conduct more research in the future. We also urge companies to more openly participate in these research projects as the sometimes observed "unwillingness" to share data negatively impacts the quality of reported research.

Limitations of this review: The main limitations of the review are bias in the selection of publications and inaccurate data extraction. To minimize the selection bias, a research protocol, selection criteria and the specific research questions were developed beforehand. Building on that we created search terms for literature selection. As other systematic literature reviews before, this review also suffers from the fact that keywords are not standardized. While we used rather broad search terms, the risk of omitting relevant literature could not be completely mitigated. To further decrease the problem of selection bias, a carefully executed multi-stage process for selecting relevant literature was chosen. The data extraction phase suffered from the fact that almost all lacked sufficient details about the design and findings of the study. Very often, the study was a "by-product" of a developed methodology or tool. However, due to the fact that only 17 studies were identified for inclusion, a manual data extraction could be performed on all articles. During the manual data extraction, we frequently found that the researchers inadequately described the set-up, issues of bias and validity were almost never addressed and applied methods were hardly ever explained. Overall, a satisfyingly presented study could not be found. Therefore, there is a possibility that the data extraction may have resulted in inaccurate results.

47.6 Conclusion

We identified 475 studies from search of the literature in electronic databases, of which 17 were found to be relevant studies of acceptable rigor, credibility, and relevance. The literature fell into three thematic groups: evaluation of out-of-the-box tools, surveys, and evaluation of one KS platform developed by the authors. We identified a number of reported challenges and benefits of the introduction of KS platforms to RE processes. However, the strength of evidence is very low, which makes it difficult to make highly convinced suggestions to industry. Therefore, practitioners are highly advised to assess the findings in careful comparison with their own situation without prematurely making generalizations.

A clear finding of this systematic review is that the number and quality of studies on the impact of KS platforms must be increased. In particular, many of the already established KS paradigms such as wikis warrant further detailed attention.

Acknowledgement The research herein is partially conducted within the competence network Softnet Austria (www.soft-net.at) and funded by the Austrian Federal Ministry of Economics (bm:wa), the province of Styria, the Steirische Wirtschaftsförderungsgesellschaft mbH. (SFG), and the city of Vienna in terms of the center for innovation and technology (ZIT). This work was supported by the project “QE LaB—Living Models for Open Systems (FFG 822740)” and partially funded by the European Commission under the FP7 project “PoSecCo” (IST 257129).

References

1. Hildenbrand FRGB, Rodriguez F, Geisser M, Berkling K, Hildenbrand T (2007) Evaluating collaboration platforms for offshore software development scenarios. In: Proceedings of the 1st international conference on software engineering approaches for offshore and outsourced development. SEAFOOD’07, Springer, Berlin, pp 96–108
2. Jiménez M, Piattini M, Vizcaíno A (2009) Challenges and improvements in distributed software development: a systematic review. Adv Soft Eng 2009:3:1–3:16
3. Sengupta B, Chandra S, Sinha V (2006) A research agenda for distributed software development. In: Proceeding of the 28th international conference on software engineering - ICSE’06, pp 73-81
4. Prikladnicki R, Nicolas Audy JL, Evaristo R (2003) Global software development in practice lessons learned. Softw Process Improv Pract 8(4):267–281
5. Sillaber C, Chimiak-Opoka J, Breu R (2012) Understanding and modeling usage decline in social networking services. In: 7th international conference on knowledge management in organizations: service and cloud computing, vol 7, pp 377–388
6. Finnegan GKB (2010) Coordinating global virtual teams: building theory from a case study of software development. Lect Notes Comput Sci 6051:281–295
7. Maccari PC Empirical analysis of a distributed software development project
8. Aurum A, Daneshgar F, Ward J (2008) Investigating knowledge management practices in software development organisations – An Australian experience. Inf Softw Technol 50(6):511–533
9. Buchan J, Ekadharmawan CH, MacDonell SG (2009) Insights into domain knowledge sharing in software development practice in SMEs. In Sulaiman S, Noor N (eds) Apsec 09:

16th Asia-Pacific software engineerig conference proceedings. IEEE Computer Society Press, pp 93–100

10. Clerc V (2008) Towards architectural knowledge management practices for global software development. In: Proceedings of the 3rd international workshop on Sharing and reusing architectural knowledge. SHARK'08, ACM, New York, NY
11. Knauss E, Boustani CE (2008) Assessing the quality of software requirements specifications. In: 16th IEEE international requirements engineering conference 2008, pp 341–342
12. Paetsch F, Eberlein A, Maurer F (2003) Requirements engineering and agile software development. WET ICE 2003. In: Proceedings. twelfth IEEE international workshops on enabling technologies: infrastructure for collaborative enterprises, Linz, 9–11 June 2003, Austria, pp 308–313
13. Sommerville I, Ransom J (2005) An empirical study of industrial requirements engineering process assessment and improvement. ACM Trans Softw Eng Methodol 14(1):85–117
14. Tiwana A (2004) An empirical study of the effect of knowledge integration on software development performance. Inf Softw Technol 46(13):899–906
15. El Emam K, Madhavji NH (1995) A field study of requirements engineering practices in information systems development. In: Proceedings of the second IEEE international symposium on requirements engineering, IEEE, pp 68–80
16. Al-Rawas A, Easterbrook S (1996) Communication problems in requirements engineering: a field study. Cogn Sci Res 31:1–2
17. Begel A, DeLine R, Zimmermann T (2010) Social media for software engineering. In: Proceedings of the FSE/SDP workshop on future of software engineering research, ACM (2010) pp 33–38
18. Hoss AM, Carver DL (2008) Innovations for requirement analysis. from stakeholders' needs to formal designs. Springer, Berlin, pp 85–102
19. Lin CYI, Ho CS (1999) A generic ontology-based approach for requirement analysis and its application in network management software. Artif Intell Eng Des Anal Manuf 13(1):37–61
20. Ratchev S, Urwin E, Muller D, Pawar KS, Moulek I (2003) Knowledge based requirement engineering for one-of-a-kind complex systems. Knowl Based Syst 16(1):1–5
21. Rus I, Lindvall M, Sinha S (2002) Knowledge management in software engineering. IEEE software 22:27–64
22. Marczak DDK (2010) Requirements-driven collaboration: leveraging the invisible relationships between requirements and people. Springer, Berlin
23. Parviainen P, Tihinen M (2011) Knowledge-related challenges and solutions in GSD. Expert Syst 23:314–341
24. Miller JA, Ferrari R, Madhavji NH (2010) An exploratory study of architectural effects on requirements decisions. J Syst Softw 83(12):2441–2455
25. Pal DDZV An industrial case study of immediate benefits of requirements engineering process improvement at the australian center for unisys software
26. Piattini GNAV (2010) A framework to improve communication during the requirements elicitation process in GSD projects
27. Dybå T, Dingsø yr T (2008) Empirical studies of agile software development: a systematic review. Inf Softw Technol 50(9–10):833–859
28. Schulte E, Davison D (2011) Active documents with org-mode. Comput Sci Eng 13(3):66–73
29. Henningsson CWH (2006) Empirical research methods in web and software engineering, Web Engineering. Springer, Berlin, Heidelberg, pp 409–430
30. Hawker S, Payne S, Kerr C, Hardey M, Powell J (2002) Appraising the evidence: reviewing disparate data systematically. Qual Health Res 12(9):1284–1299
31. Damian D (2007) Stakeholders in global requirements engineering: lessons learned from practice. IEEE Softw 24(2):21–27
32. Paech ESAB (2007) A field study of the requirements engineering practice in Australian software industry
33. Fowler D, Crowder R, Guan T, Shadbolt N, Wills G (2010) Requirements for semantic web applications in engineering. In: Proceedings of International Design Engineering Technical

Conferences & Computers and Information in Engineering Conference, ASME 2010, Montreal Quebec, canada, 15–18 Aug 2010, pp 1–9

34. Hong Q, Kim S, Cheung S, Bird C (2011) Understanding a developer social network and its evolution. In: 2011 27th IEEE international conference on software maintenance (ICSM). number 612108, pp 323–332
35. Leuf B, Cunningham W (2001) The Wiki way: quick collaboration on the web. Addison-Wesley, New York
36. Ras E, Rech J (2009) Using wikis to support the net generation in improving knowledge acquisition in capstone projects. J Syst Softw 82(4):553–562
37. Richards D (2009) A social software/web 2.0 approach to collaborative knowledge engineering. Inf Sci 179(15):2515–2523
38. Berkling K, Geisser M, Hildenbrand T, Rothlauf F, Rothlauf KBGH (2007) Offshore software development: transferring research findings into the classroom. SEAFOOD'07. Springer, Berlin
39. Stettina CJ, Heijstek W (2011) Necessary and neglected?: an empirical study of internal documentation in agile software development teams. In: Proceedings of the 29th ACM international conference on design of communication. SIGDOC'11, New York, NY, ACM, pp 159–166
40. Rech J, Bogner C, Haas V (2007) Using wikis to tackle reuse in software projects. IEEE Softw 24(6):99–104
41. Luna-Reyes LF, Black LJ, Cresswell AM, Pardo TA (2008) Knowledge sharing and trust in collaborative requirements analysis. Syst Dyn Rev 24(3):265–297
42. Ovaska E, Evesti A, Henttonen K, Palviainen M, Aho P (2010) Knowledge based quality-driven architecture design and evaluation. Inf Softw Technol 52(6):577–601
43. Mishra SA, Aydin S, Mishra D (2009) A tool to enhance cooperation and knowledge transfer among software developers. CDVE 2009:257–260
44. Kabanda S, Adigun, M (2006) Extending model driven architecture benefits to requirements engineering. In: Proceedings of the 2006 annual research conference of the South African institute of computer scientists and information technologists on IT research in developing countries. SAICSIT'06, Republic of South Africa, South African Institute for Computer Scientists and Information Technologists
45. Minocha S, Thomas PG (2007) Collaborative learning in a wiki environment: experiences from a software engineering course. New Rev Hypermedia Multimedia 13(2):187–209